Introducing **Live Contracts** @ Stanford CodeX

What is a **Live Contract**?

A Live Contract is an agreement that is formalized as

Finite State Machine

LegalThings **One** runs Live Contracts and stores information in a provable, immutable way on public decentralized storage.

Defusing power

1. Empower self-reliance
2. Mitigate fraud
3. Reduce workload and friction

Empowering self-reliance

- Understand the rules
- Access to information
- Power to execute

Mitigating fraud and reduce workload

bureaucracy × fraud = constant

Formalizing an agreement

Paper Contracts

- Humans only
- Ambiguous
- Stateless
- Private

Natural language

Live Contracts

- Humans & computers
- Unambiguous past, ambiguous present
- Deterministic state
- Public & private

Finite state machine

Smart Contracts

- Computers only
- Unambiguous
- Stateful
- Public

Turing complete

Smart Contracts vs Live Contracts

Who will determine these unambiguous facts?

Formalizing an agreement, a simple example

Who:	Jake (16)
Side job:	Mowing lawn of his Uncle Mike
Pay:	10 USD
Conditions:	Grass < 2 inches + don't damage lawn or gnomes
Gnomes:	20
Fine:	30 USD / damaged gnome
Frequency:	once a week
Duration:	infinite
Arbiter:	Frank (Jake's father)

Example - Paper contract

"Jacob M.S. McGill (hereafter "Jake") and Michael F. McGill (hereafter "Mike") agree that Jake will mow Mike's lawn once a week to a length of 2 inches (hereafter "the Work"), for which Mike will pay Jake USD 10, unless Jake damages the lawn or any of the ornamental garden gnomes (hereafter "Gnomes") present on the lawn in the course of the Work. Jake will reimburse Mike with USD 30 for each Gnome damaged. Disputes will be settled by Frances W. McGill (hereafter "Frank")."

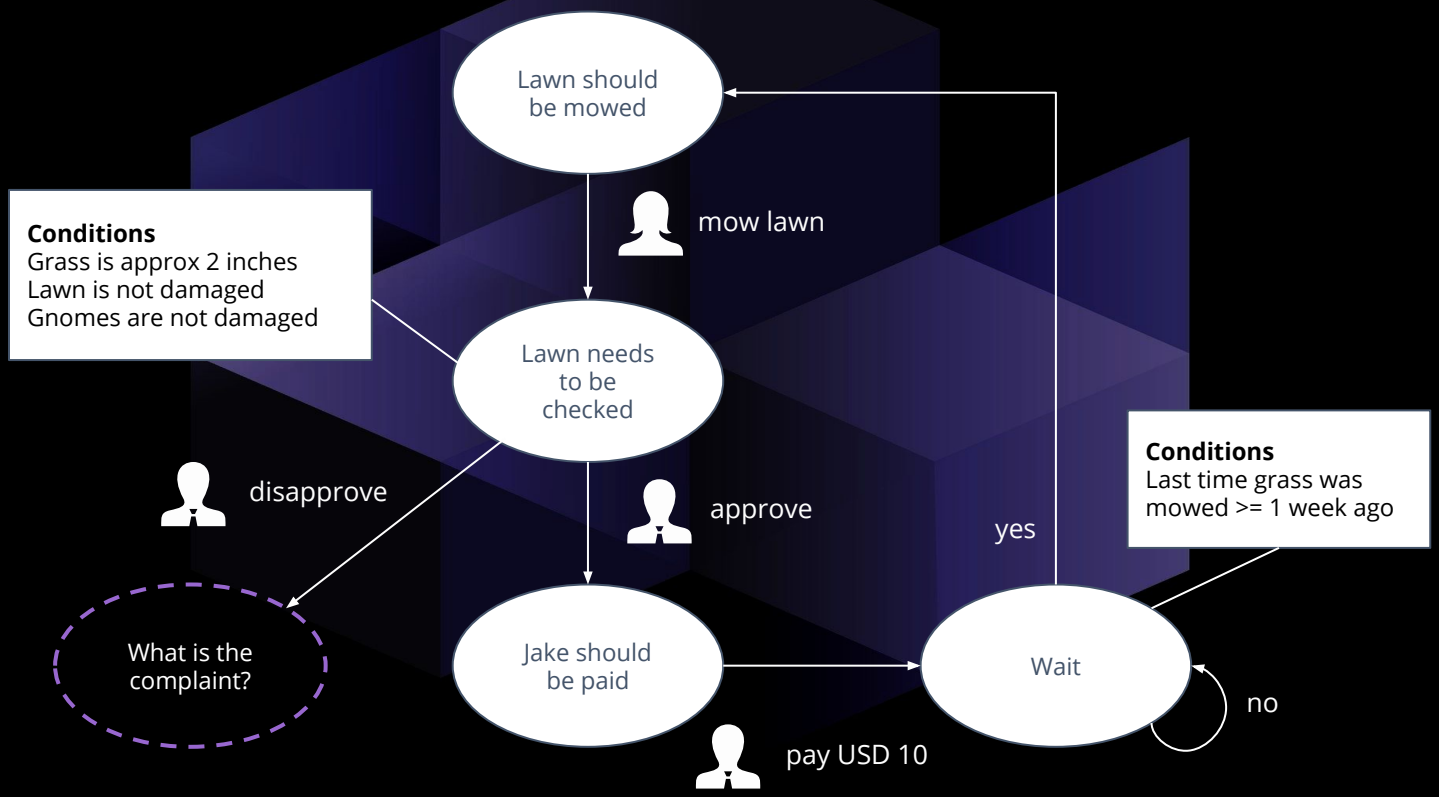
Example - Smart contract

```
contract AccessRestriction {
    address public owner = msg.sender;
    bool fair = true;

    function setBonus(uint8 _bonus) public onlyBy(owner) {
        fair = _bonus > 0;
    }

    modifier onlyBy(address _account)
    {
        require(msg.sender == _account);
        _;
    }

    modifier costs(uint _amount)
    {
        require(msg.value >= _amount);
        _;
        if (msg.value > _amount)
            msg.sender.send(msg.value - _amount);
    }

    modifier mustBeFair() {
        require(fair);
        _;
    }
}

contract MowTheLawn is WithdrawalContract, AccessRestriction {
    address public employer = msg.sender;
    address public employee;
    address public arbiter;
    bool isSigned = false;
    uint pending = 0;
    bool hasConflict = false;

    function MowTheLawn(address _employee, address _arbiter) public payable costs(2 ether) {
        employee = _employee;
        arbiter = _arbiter;
    }

    function accept() public payable costs(2 ether) onlyBy(employee) {
        isSigned = true;
    }

    function submitWork() public onlyBy(employee) onlyIfSigned() mustBeFair() {
        pending++;
    }

    function approveWork() public onlyBy(employer) onlyIfSigned() mustBeFair() {
        employee.transfer(pending * 650 finney);
        pending = 0;
    }

    function disputeWork() public onlyBy(employer) onlyIfSigned() mustBeFair() {
        hasConflict = true;
    }

    function resolve(bool _approve) public onlyBy(arbiter) onlyIfSigned() mustBeFair() {
        if (_approve) {
            approveWork();
        } else {
            pending = 0;
        }
    }

    modifier onlyIfSigned() {
```

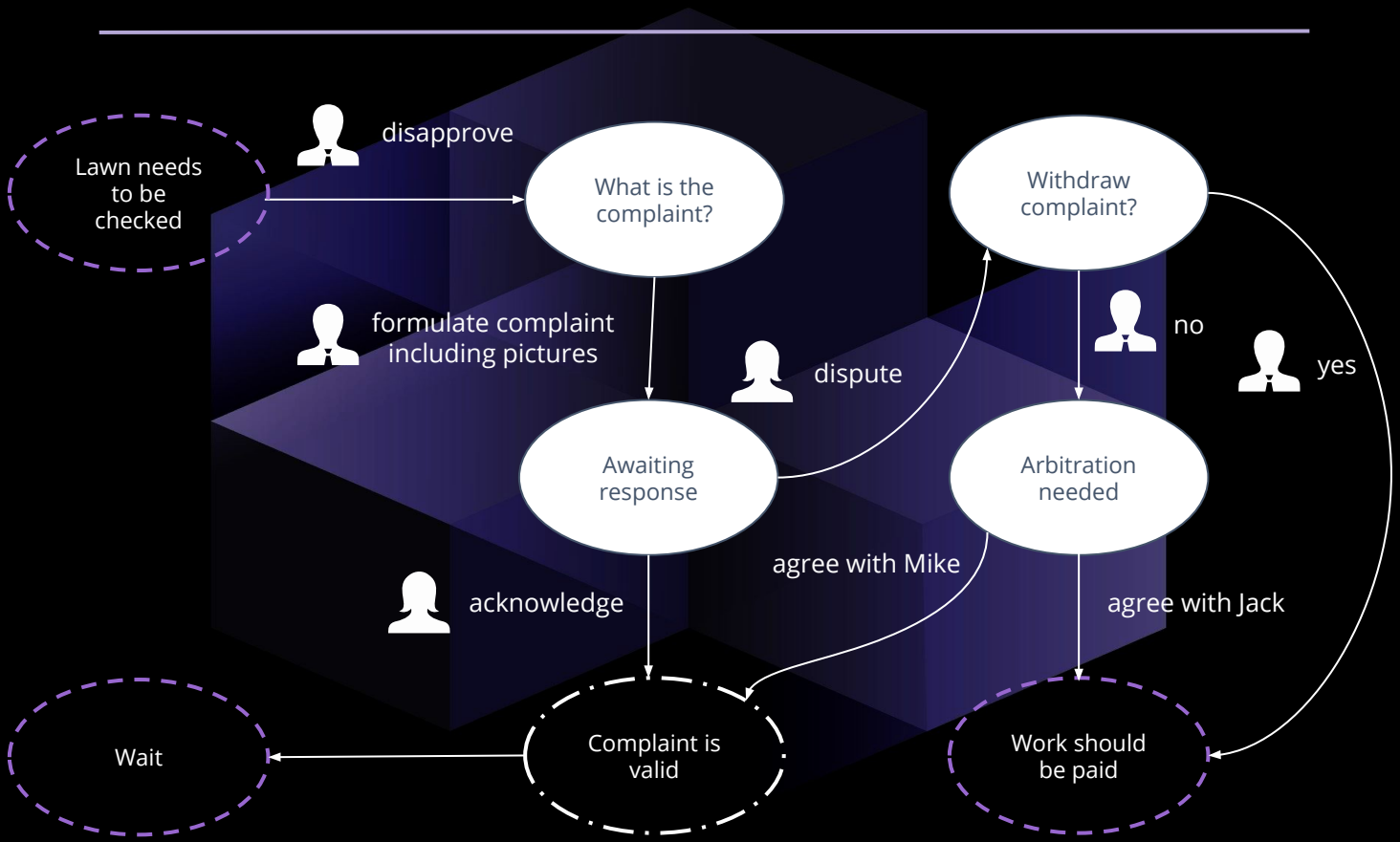
Example - Smart contract

```
function MowTheLawn(address _employee, address _arbiter) public payable costs(2 ether) {
    employee = _employee;
    arbiter = _arbiter;
}

function accept() public payable costs(2 ether) onlyBy(employee) {
    isSigned = true;
}
```

Example - Live Contract



Example - Live Contract



Demo of Live Contract

Try a Live Contract? Go to Livecontracts.io

Thank you for your attention

Time for **Q&A**

@getlegalthings

@livecontracts

www.legalthings.io

www.livecontracts.io